

# Machine Learning Sentiment Prediction based on Hybrid Document Representation

Panagiotis Stalidis, Maria Giatsoglou, Konstantinos Diamantaras<sup>a</sup>

*Department of Information Technology,  
TEI of Thessaloniki, GR-57400 Thessaloniki, Greece*

George Sarigiannidis, Konstantinos Ch. Chatzisavvas<sup>b</sup>

*mSensis S.A., VEPE Technopolis, Bld C2, GR-57001 Thessaloniki, Greece*

## Abstract

Automated sentiment analysis and opinion mining is a complex process concerning the extraction of useful subjective information from text. The explosion of user generated content on the Web, especially the fact that millions of users, on a daily basis, express their opinions on products and services to blogs, wikis, social networks, message boards, etc., render the reliable, automated export of sentiments and opinions from unstructured text crucial for several commercial applications. In this paper, we present a novel hybrid vectorization approach for textual resources that combines a weighted variant of the popular Word2Vec representation (based on Term Frequency-Inverse Document Frequency) representation and with a Bag-of-Words representation and a vector of lexicon-based sentiment values. The proposed text representation approach is assessed through the application of several machine learning classification algorithms on a dataset that is used extensively in literature for sentiment detection. The classification accuracy derived through the proposed hybrid vectorization approach is higher than when its individual components are used for text representation, and comparable with state-of-the-art sentiment detection methodologies.

---

<sup>a</sup> kdiamant@it.teithe.gr

<sup>b</sup> kchatz@msensis.com

## I. INTRODUCTION

*Sentiment Analysis* (often referred to as *Opinion Mining*) involves the application of Natural Language Processing, text analytics, and computational linguistics methods and tools in order to extract subjective information from text. It is applied on a variety of textual sources that typically carry peoples' opinions, emotions, and evaluations on specific *entities*, such as individuals, events or topics (for instance, reviews of movies, books, products, etc.). Sentiment Analysis aims mainly at *identifying the sentiment content* of the textual resource under inspection, and subsequently *estimating its polarity* (positive/negative). Related tasks are *question answering* (recognizing opinion oriented questions) and *summarization* (accounting for multiple viewpoints). The sentiment analysis process can be carried out on different granularity levels: at the level of the *document*, *sentence* or *aspect*. *Document-level* Sentiment Analysis aims at classifying a multi-sentence document in terms of the polarity of the opinion expressed within it. This approach assumes that the whole document expresses opinions on a single entity (e.g. a single topic), and is not applicable to documents that refer to multiple entities [1]. *Sentence-level* or *phrase-level* Sentiment Analysis seeks to classify the sentiment expressed in a single sentence, and characterize the sentence as positive, negative, or neutral. Liu 2012 argues that there is no fundamental difference between document- and sentence-level classification, since sentences are simply short documents. *Aspect-level* Sentiment Analysis aims to extract sentiments expressed with respect to certain aspects of the entities. This is necessary in many applications, especially when products or services are evaluated. For example, the sentence “*The voice quality of this phone is not good, but the battery life is long*”, expresses a negative sentiment towards the quality of the product but a positive sentiment regarding its durability. Typically, Sentiment Analysis is applied on text corpora containing product reviews. However, it can also be applied on news articles [3] or on documents which are generated in social networks and microblogging sites. In the latter case the aim is to extract the public opinion on various topics ranging from stock markets [4] to political debates [5].

The two main approaches for automated sentiment extraction are the *Lexicon-based* approach and the *Machine Learning* approach.

Lexicon-based Sentiment Analysis involves the estimation of the sentiment of a textual resource from the semantic orientation of the words or phrases it includes [6]. It has primarily focused on using adjectives as indicators of the semantic orientation of text [7–10] or adjectives and adverbs combined [11]. Lists of such terms together with their sentiment orientation values are collected into dictionaries, and then the sentiment of a textual resource is estimated through the aggregation

of the values of the lexicon terms found in the document. Dictionaries for lexicon-based approaches can be created manually [12, 13] or automatically [6, 7, 14] using seed words to expand the list of words. Since one of the steps in lexicon-based approaches is to annotate text with part-of-speech tags, the aspects that the sentiment is targeted upon can be easily detected.

Machine Learning (ML) Sentiment Analysis involves supervised training of classifier models using labeled instances of documents or sentences [1]. A number of features are extracted from the text, such as words (unigrams), groups of successive words (n-grams), or even numeric features, such as the number of nouns, verbs, or adjectives, included in the text. Then, these features are represented by vectors which are fed into suitable ML algorithms for deriving the classification model. Many text classifiers have been proposed in literature, including Decision Trees, Naïve Bayes, Rule Induction, Neural Networks, Clustering K-Nearest Neighbours, and Support Vector Machines (SVM). One of the earliest works following the ML approach has been proposed by Pang et al. 2002. The authors employed a bag-of-features representation approach on movie reviews using unigrams, bigrams, adjectives and the position of words as features. In order to predict sentiment they used a number of standard ML classifiers concluding that the best performance is achieved when the unigrams are used in combination with an SVM classifier. Whitelaw et al. 2005 augmented the bag-of-words approach with shallow parsing that extracts opinion phrases, classified into attitude types derived from appraisal theory [16]. An SVM was applied on the feature vectors comprising word frequencies and the percentage of appraisal groups with particular orientations. They achieved 90.2% accuracy classifying the movie reviews corpus introduced by Pang et al. 2002. Ye et al. 2009 used sentiment classification techniques for classifying online reviews for travel destinations. They used the frequency of words instead of word presence to represent a document, and the Information Gain measure to select the feature set. They compared the performance of SVM, Naïve Bayes and a character-based N-gram model on the classification of travel destination reviews. SVM was found to outperform the other two classifiers with an accuracy peak of  $\sim 86\%$  when the training corpora contained 700 reviews. Prabowo and Thelwall 2009 used a hybrid classification scheme employing a cascade of rule-based classifiers in combination with an SVM-based approach to separate the two classes (positive/negative). The hybrid classifier was tested on movie reviews, product reviews and social media comments (from MySpace) yielding an F-measure score between 72.77% and 90%.

This paper introduces a hybrid method that combines both lexicon-based features and machine learning for sentiment classification. We propose a novel document representation approach that combines the Bag-of-Words representation with a TF-IDF weighted Word2Vec representa-

tion together with a vector of lexicon-based sentiment values. Several supervised machine learning methods are then trained using the derived representation vectors as inputs and the polarity values as targets, and are comparatively evaluated based on their performance. For our simulation experiments we selected a well-known movie reviews corpus, so that our results can be easily compared against other methods.

The rest of the paper is organized as follows. In Section II we describe the sentence vector representation methodology, and in Section III we outline the machine learning models used for training the system. Then, in Section IV we evaluate the performance of our proposed approach based on a series of experiments, and compare it with competitive methods. Finally, in Section V we discuss the benefits and limitations of the proposed approach.

## II. VECTOR REPRESENTATION OF SENTENCES

Efficient vector representations of documents is essential to any algorithm that performs Sentiment Analysis. Typically, such representations have one of the two following properties:

- (a) ability to capture semantic similarities between documents, i.e. paragraphs of similar meaning are coded by vectors which are close to each other in Euclidean space;
- (b) ability to capture sentiment polarity or emotional content.

Unfortunately, representations that have the first property fail to capture sentiment. On the other hand, lexicon-based approaches that have the second property fail to capture semantic similarity between documents. The motivation behind our proposed approach is our belief that a combination of both properties is beneficial, if not necessary, for sentiment analysis.

The first property facilitates the improvement of the generalization capability for any ML model that uses these representations to classify documents in terms of their sentiment polarity or emotional content. The Bag-of-Words (BoW) approach [19], for instance, represents a document using the frequency of occurrence of each word, disregarding the word order, and often considering only terms from a given dictionary or corpus. Semantic similarity between documents can be captured by comparing their term frequency profiles (i.e. representation vectors). However, BoW fails to capture similarity, in the case of synonyms. For example, the sentences: “*He came late to school*” and “*He delayed coming to class*” have quite different BoW representations although they have very similar meanings. The Word2Vec model [20], described in more detail below, tries to overcome this limitation by representing words with vectors such that words appearing frequently in

similar contexts have close representations in the Euclidean space. Even so, such semantics-based representations alone, ignore the sentimental value of the terms included in the document.

The second property can be achieved, for example, by the use of a lexicon where terms are assigned scores corresponding to their sentiment orientation. Early approaches that are characterized by only this property are called *lexicon-based* methods [12]. They suffer from low coverage, i.e. the fact that many sentences may not contain any terms from the lexicon and, thus, their sentimental orientation cannot be evaluated. To mitigate the disadvantages of both approaches, we propose to use lexicon-based representations in complementary fashion to the semantics-based representations. Thus, the combined representation will have both properties described above.

Motivated by the above discussion, we propose the **Hybrid Weighted Word2Vec** (HWW2V) text representation approach, which is a concatenation of:

- (a) *Bag-of-Words* representation,
- (b) *Weighted Word2Vec* representation, and
- (c) *Sentiment lexicon-based* representation.

The next subsections provide the details for each individual representation approach.

### A. Morphological Processing

Prior to extracting the features that are fed into the classifier, a number of preprocessing steps are taken as described below:

- *Tokenization*: Split the text into smaller parts mainly words.
- *Contractions*: Replace contractions with two tokens, e.g., *can't* is replaced by *can* and *not*.
- *Negations*: Negations are treated by adding artificial words [21]. For example, if a word  $x$  is preceded by a negation word, then a new feature **negated\_x** is created. Since, as suggested by [1] the scope of negation cannot be properly modelled, every word is replaced until the end of the sentence.
- *Stopword Removal*: Remove common words which convey no sentiment or semantics, such as “a”, “the”, “to”, etc.

## B. Bag-of-Words Representation

The dictionary used in the BoW representation is the set of all words in the document corpus after removing the stop words. Furthermore, all words were turned to lower-case but no lemmatization or stemming was applied. We used the Term Frequency-Inverse Document Frequency (TF-IDF) representation of each document where the term frequency was forced to be binary (i.e. a given word is present or not) and only the document frequency varied.

## C. Weighted Word2Vec Representation

Mikolov et al. 2013 proposed the *Word2Vec* model that embeds semantic information in a vector space representation of words. Its key idea is the capturing of the context of words by using ML approaches such as Neural Networks. Word2Vec predicts the occurrence of a word given a window of previous and subsequent words. It incorporates two different architectures for the computation of vector representations of words from large datasets:

- (a) Continuous Bag-of-Words model (CBOW): predicts a word when the surrounding words are given. It is much faster than the Skip-gram model and slightly more accurate for frequent words;
- (b) Skip-gram model (SG): predicts a window of words when a single word is known. It operates well with a small amount of training data representing accurately even rare words and phrases.

The learned vectors explicitly encode many linguistic patterns, for example, vectors encoding words with similar meanings are close in the Euclidean space. Also, many of these patterns can be represented as linear translations [22]. Following these successful techniques, research interest has been extended to the development of models that go beyond word-level to achieve sentence-level representations [23–28]. Probably, the most straightforward approach to derive document-level representations is to simply take the average of the vectors of all words contained in the document, according to the Word2Vec embedding (i.e. *mean Word2Vec*). Doc2Vec is a more sophisticated approach, which modifies the original Word2Vec algorithm to the unsupervised learning of continuous representations for larger blocks of text, such as sentences, phrases, paragraphs or entire documents [28].

*a. Proposed approach:* The *Weighted Word2Vec*, introduced here, is a modification of *mean Word2Vec*, where we represent a document by the *weighted average* of the representations of all its words as they are obtained by the Word2Vec model. For each word  $i$  in the dictionary we compute its vector representation  $\mathbf{v}_i$  by using the *CBOW* model and we construct a *dictionary matrix*

$$\mathbf{V} \in \mathbb{R}^{N \times B},$$

where  $N$  is the dimensionality of each word representation and  $B$  is the size of the dictionary. Each term *weight* should reflect its "importance" within the document, and thus it is directly proportional to its *Term-Frequency* normalized by the *Inverse-Document-Frequency*, i.e. the TF-IDF measure. For each document  $j$  and word  $i$  we calculate the TF-IDF, as

$$w_{ij} = f_{ij} \log \frac{D}{D_i} \quad (1)$$

where  $f_{ij}$  is the frequency of the  $i$ -th word in document  $j$ ,  $D$  is the total number of documents in the corpus, and  $D_i$  is the number of documents containing the  $i$ -th word. Then the vector representation  $\mathbf{s}_j$  of the  $j$ -th document is the sum of the words that appear in it weighted by their TF-IDF values:

$$\mathbf{s}_j = \sum_i w_{ij} \mathbf{v}_i \quad (2)$$

#### D. Sentiment Lexicon-based Representation

The Word2Vec-based representations described in the previous paragraphs do not contain sentiment information about the words included in the document. We wish to study the advantages of augmenting this representation by including the semantic orientation of words based on a lexicon that associates a set of terms with their sentimental polarity expressed as a numerical value. This approach, of course, assumes that semantic orientation is independent of context. The lexicon can be created either manually, such as the General Inquirer lexicon [29], or semi-automatically, utilizing resources like WordNet [9, 30]. Our implemented model makes use of the semi-automatically created sentiment lexicon SentiWordNet (SentiWordNet) which has been applied in different opinion related tasks, i.e., for subjectivity analysis and sentiment analysis, with promising results.

SentiWordNet extends the WordNet lexicon of synonyms adding sentiment values for each synset (i.e., sets of cognitive synonyms) in the collection by means of a combination of linguistic and statistic classifiers. SentiWordNet assigns to each synset of WordNet three sentiment scores, based on its *positivity*, *negativity* and *objectivity*, respectively, with the sum of these scores being always 1.

The model we implemented using SentiWordNet, calculates a positive, objective and negative score for each token encountered in the corpus, by averaging the values of all synsets in SentiWordNet that have the same text representation. Tokens that are used after a negation, i.e. **NOT\_good**, have their polarity reversed. In order to anticipate words that are not included in the dictionary, a fourth feature was added, denoting unknown sentiment. The sentiment scores for all tokens (including their reversed forms) are represented in the *sentiment matrix*.

Since a token generally does not pass its sentiment in another sentence (apart from the one it resides in), we first split each review in sentences. Then, for each sentence we form a vector as in a BoW model, and we aggregate its overall sentiment representation by calculating the dot product of the sentiment matrix with the sentence vector. To aggregate the sentiment at the document level, we calculate the mean sentiment representation across all the document's sentences. This process extracts four features for each document:

1. a positive sentiment value,
2. an objective value,
3. a negative sentiment value, and
4. a value indicating the percentage of words that are not included in SentiWordNet.

*Example:*

Text	Sentence Vector	Positive	Objective	Negative	Unknown
-	1	0	0	0	1
.	1	0	0	0	1
bad	1	0	0.2	0.8	0
between	1	0	1	0	0
dead	1	0.1	0.3	0.6	0
man	1	0	1	0	0
room	1	0	1	0	0
smell	1	0.2	0.6	0.2	0
smt	1	0	0	0	1
so	2	0	1	0	0
towels	1	0	1	0	0
wardrobe	1	0.1	0.9	0	0
wet	1	0.1	0.8	0.1	0



The inner product of the sentence vector and the sentiment matrix

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0.2 & 0.8 & 0 \\ 0 & 1 & 0 & 0 \\ 0.1 & 0.3 & 0.6 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0.2 & 0.6 & 0.2 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0.1 & 0.9 & 0 & 0 \\ 0.1 & 0.8 & 0.1 & 0 \end{pmatrix} = \begin{pmatrix} 0.036 & 0.671 & 0.121 & 0.171 \end{pmatrix}$$

Thus, the vector representation of the text is

Positive	Objective	Negative	Unknown
0.036	0.671	0.121	0.171

### III. MACHINE LEARNING CLASSIFICATION MODELS

The document representations resulting from the concatenation of the BoW, Weighted Word2Vec, and Sentiment Lexicon-based vectors are used to train a supervised classifier to learn the documents' sentiment polarity. In this work we have evaluated a number of popular machine learning methods as detailed below.

#### A. Naïve Bayes

The probability of a document  $d$  being in class  $c$  is computed as

$$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c) \quad (3)$$

where  $(t_1, t_2, \dots, t_{n_d})$  are the tokens in the vocabulary that appear in  $d$  and  $n_d$  is the number of the tokens. Classification is performed by choosing the maximum a posteriori (MAP) estimate:

$$C_{\text{map}} = \arg \max_{c \in C} P(c|d) \quad (4)$$

In order to avoid floating point underflow we use the log probability

$$\log P(c|d) = \log P(c) + \sum_{1 \leq k \leq n_d} \log P(t_k|c)$$

Ensemble estimates are used for the values  $P(c)$  and  $P(t_k|c)$ . One complication is that if a term  $t_k$  never appears in class  $c$ , the value  $P(t_k|c)$  is zero and due to the multiplicative rule in Naïve Bayes, the overall probability will be zero, independent of how strong evidence other terms may offer. We avoid this problem by adding 1 to all term counts, thus modifying the equation to estimate the conditional probability to

$$P(t_k|c) = \frac{N_{ct} + 1}{\sum_{t' \in V} (N_{ct'} + 1)} = \frac{N_{ct} + 1}{(\sum_{t' \in V} N_{ct'}) + B} \quad (5)$$

where  $B$  is the number of terms in the vocabulary  $V$ .

## B. Maximum Entropy

Considering any feature  $x_i(d, c)$  of a document  $d$  in class  $c$ , the maximum entropy principle restricts the model distribution to have the expected value of this feature equal to the average obtained from the training data,  $D$

$$\frac{1}{|D|} \sum_{d \in D} x_i(d, c) = \sum_d P(d) \sum_c P(c|d) x_i(d, c) \approx \frac{1}{|D|} \sum_{d \in D} \sum_c P(c|d) x_i(d, c) \quad (6)$$

When constraints are imposed in this fashion, it is guaranteed that a unique distribution exists that has maximum entropy [31]. Moreover, it has been shown [32] that the distribution is always of the exponential form

$$P(c|d) = \frac{1}{Z(d)} \exp \left( \sum_i \lambda_i x_i(d, c) \right) \quad (7)$$

where  $\lambda_i$  is a parameter to be estimated and  $Z(d)$  is the *partition function* that ensures that  $P(c|d)$  is a proper probability. When the constraints are estimated from labeled training data, the solution to the ME problem is also the solution to a dual maximum likelihood problem for models of the same exponential form. Additionally, it is guaranteed that the likelihood surface is convex, having a single global maximum and no local maxima. A possible approach for finding the maximum entropy solution would be to guess any initial exponential distribution of the correct form as a starting point and then perform hill climbing in likelihood space. Since there are no local maxima, this will converge to the maximum likelihood solution for exponential models, which will also be the global maximum entropy solution.

### C. Support Vector Machines

Support Vector Machines (SVM) [33] are called maximum margin classifiers, since, in the case of a separable problem, the obtained separating surface maximizes the distance of the closest pattern from it. This is achieved by solving the following optimization problem: Maximize

$$L(\mathbf{a}) = \sum_i a_i - \frac{1}{2} \sum_{i,j} a_i a_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \quad (8)$$

under the conditions

$$0 \leq a_i \leq C \quad \text{and} \quad \sum_i a_i y_i = 0$$

where  $y_i \in \{-1, 1\}$  is the class label for document  $i$ ,  $\mathbf{x}_i$  is the vector representation of document  $i$ ,  $C$  is a user parameter specifying how important the misclassification penalty is, and  $K(\cdot, \cdot)$  is a kernel function (polynomial, Radial Basis Function (RBF), hyperbolic tangent, etc.). After solving the above optimization problem the optimal classifier is

$$f(\mathbf{x}) = \sum_i a_i y_i K(\mathbf{x}_i, \mathbf{x}) \quad (9)$$

## IV. EXPERIMENTS

### A. Experimental Methodology

In all experiments reported below we evaluated the classification models using a 10-fold cross-validation protocol leaving 10% of the patterns out for testing.

Figure 1 describes the steps used in our experiments. First, the documents are pre-processed in order to handle negations, contractions, stop-words, etc., as described in the next subsection. Then the processed documents are represented in three different ways using the BoW, Weighted Word2Vec, and Sentiment lexicon-based vectors as described in Section II. Additionally, all three vectors are combined together in the “HWW2V” representation.

All representations, the three individual and the hybrid one, were tested separately using different types of shallow learning classifiers: Naïve Bayes, Maximum Entropy, SVM with linear kernel, and SVM with RBF kernel (see Section III) based on the accuracy of the classification model on the sentiment detection task. The purpose of this experiment is to investigate the potential improvement offered by the hybrid representation.

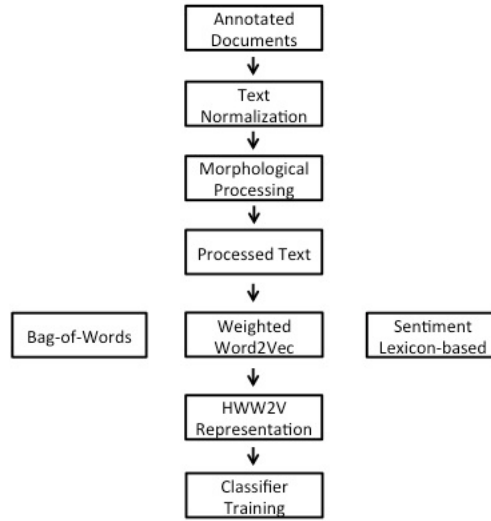


FIG. 1. Experimental set-up

### 1. The Movie Reviews Dataset

To test our method we use the popular RT movie reviews data set[34] introduced by Pang and Lee 2005. It is a collection of 10662 processed sentences/snippets from movie reviews written in English by users in the Rotten Tomatoes website. The collection contains 5331 positive and 5331 negative reviews.

### 2. Machine Learning Algorithms

As mentioned above, we employed both shallow and deep learning models for the classification stage. In the case of shallow models the following popular algorithms were tested: (a) Naïve Bayes, (b) Maximum Entropy, (c) SVM with linear kernel, and (d) SVM with RBF kernel. For all of the above methods we used Python’s *scikit-learn* machine learning implementation[36]. For Naïve Bayes, we used the MultinomialNB class which implements the Naïve Bayes algorithm for multinomially distributed data, and is one of the two classic Naïve Bayes variants used in text classification. For Maximum Entropy, we used the Logistic Regression class with the liblinear solver. In both cases, we examined if scaling the input vectors to unit form is beneficial for the prediction process. We also examined how the removal of terms with high and low document frequencies affects the accuracy of the classifiers. For SVM, we used the SVC class implementation with both a linear kernel and an RBF kernel. In the case of the RBF kernel, two parameters must

TABLE I. Classification performance using different document representations: (a) only Sentiment Lexicon-based vectors, (b) only Weighted Word2Vec vectors with dimension=100 (top row) and dimension=300 (bottom row), (c) only BoW vectors, and (d) HWW2V representation with Weighted Word2Vec vectors of dimension=100 (top row) and dimension=300 (bottom row). All numbers in parentheses indicate execution times in seconds for one fold of the cross-validation experiment.

Method	Sentiment Lexicon-based	Weighted Word2Vec	BoW	HWW2V
Naïve Bayes	0.570 (0.003)	0.541 (0.013) <b>0.727</b> (0.030)	0.781 (0.012)	0.781 (0.034) <b>0.785</b> (0.067)
Maximum Entropy	0.582 (0.009)	0.623 (0.707) 0.772 (40.6)	0.773 (0.095)	0.784 (0.842) <b>0.791</b> (125.7)
SVM (linear)	0.582 (0.254)	0.620 (2.604) 0.772 (349.8)	0.772 (0.299)	0.783 (2.590) <b>0.792</b> (376.1)
SVM (RBF)	0.582 (5.02)	0.629 (22.54) 0.777 (136.7)	0.776 (28.21)	0.783 (62.69) <b>0.796</b> (3,006.5)

be considered,  $C$  and  $\gamma$ . Parameter  $C$ , common to all SVM kernels, trades-off misclassification of training examples against simplicity of the decision surface. A low  $C$  makes the decision surface smooth, while a high  $C$  aims at classifying all training examples correctly. Parameter  $\gamma$  defines how much influence a single training example has. The larger  $\gamma$  is, the closer other examples must be to be affected.

The performance of the models is evaluated based on the average test-set accuracy after 10-fold cross-validation experiments.

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

where TP and FP are the numbers of true and false positives, and TN and FN are the numbers of true and false negatives, respectively.

## B. Experimental Results

The results of our experiments using the shallow learning classifiers are summarized in Table I.

The results achieved using only the vectors obtained by the sentiment lexicon (Sentiment lexicon-based representation) are presented in the second column of Table I. The obtained accuracies are low, regardless of the classifier, implying that the features are not very successful in capturing the

document sentiment polarity. This can be explained by the fact that SentiWordNet is a general sentiment dictionary which captures an overall (broad) sentiment for each word.

The third column of Table I presents the results obtained using only the Weighted Word2Vec representation vectors as features. We note that the SVM model with RBF kernel slightly outperforms the other classifiers obtaining accuracy around 78% for vectors with dimension 300. The dimensionality of the vectors is an issue since the increase of the dimensions from 100 to 300 results to a substantial improve of the accuracy (14.8%-18.6%). Almost the same results are obtained using the BoW approach for representing the sentences. In this case, the Naïve Bayes classifier yields the best performance with 78% accuracy while the SVM/RBF model is a close second. Both BoW and Weighted Word2Vec representations seem to offer the same discriminative power regarding sentiment polarity.

The HWW2V representation yields the results displayed in the last column of Table I. It is clearly seen that HWW2V assists all four classifiers achieve better performance compared to using each representation separately. The best performance is achieved for the SVM model with RBF kernel, obtaining accuracy slightly lower than 80%. For each classifier the performance is improved between 0.5% (in the case of Naïve Bayes) and 2.5% (in the case of SVM with linear kernel) compared to the second best approach (BoW representation).

Our initial hypothesis was that lexicon-based features can not accurately capture refined characteristics and contextual cues that are inherent in the human language, mainly because people often express their emotions and opinions in subtle ways. Thus, we proposed a hybrid representation approach that combines sentiment lexicon-based features with word embedding-based approaches (such as Weighted Word2Vec), which try to capture semantic and syntactic features of words out of document collections in a language independent process. Our experiments showed that hybrid approaches can take advantage of both lexicon-based features and word embedding learning approaches to derive more accurate sentiment prediction results.

## V. DISCUSSION

We propose a hybrid methodology that combines TF-IDF weighted Word2Vec representation and BoW representation with sentiment lexicon-based values for sentiment analysis in unstructured text. The word embedding representation, based on Word2Vec, provides semantic and syntactic coding, refined further in Weighted Word2Vec with TF-IDF weighting that augments the value of frequent words. The Sentiment Lexicon-based representation provides sentiment and emotion

information for the document under inspection. The hybrid approach monetizes the merits of both representations, and it is customizable for any language as long as a sentiment lexicon is available.

Our methodology is assessed through the application of several classifiers on the *MOVIES* dataset that has been used extensively in literature for the evaluation of the various methodologies proposed for sentiment analysis. In Table V we summarize the results of the application of different methods along with the performance of our approach, HWW2W.

TABLE II. Comparison of different methods on the *MOVIES* dataset. All results in this table, except for the last row, are taken from [37].

Method	Test Accuracy	Reference
CNN-rand	0.761	
CNN-static	0.810	[37]
CNN-non-static	0.815	
CNN-multichannel	0.811	
RAE	0.777	[38]
MV-RNN	0.790	[39]
CCAE	0.778	[40]
NVSVM	0.794	[41]
MNB	0.790	
G-dropout	0.790	[42]
F-dropout	0.791	
Tree-CRF	0.773	[43]
Sent-Parser	0.795	[44]
<b>HWW2V with SVM (RBF)</b>	<b>0.796</b>	

Some details on the methods presented in Table V are provided below. In [37] the authors report a series of experimental results using Convolutional Neural Networks (CNN) trained on top of pre-trained word vectors for sentence-level classification tasks. In [38] a novel machine learning framework based on Recursive Autoencoders (RAE) for sentence-level prediction of sentiment label distributions, is introduced. The method supports learning vector space representations for multi-word phrases, without using any pre-defined sentiment lexica or polarity shifting rules. Also, in [39] a Recursive Neural Network model (RNN) is introduced that learns compositional vector representations for phrases and sentences of arbitrary syntactic type and length. Herman and Blunsom 2013 draw upon advances in the learning of vector space representations of sentential semantics and the transparent interface between syntax and semantics provided by Combinatory Categorical Grammar

(CCG), and they introduce Combinatory Categorical Autoencoders (CCAE). This model leverages the CCG combinatory operators to guide a nonlinear transformation of meaning within a sentence. They use this model to learn high dimensional embeddings for sentences and evaluate them in a range of tasks, demonstrating that the incorporation of syntax allows a concise model to learn representations that are both effective and general. Wand and Manning 2012 proposed simple but novel Naïve Bayes and SVM variants as feature values (MNB and NVSVM, respectively), which performed well across tasks and datasets, sometimes providing new state-of-the-art performance levels. In [42] the dropout training model [45] was further improved by randomly dropping out (zeroing) hidden units and input features during training of neural networks. This process speeds up training without losing classification performance. Nakagawa et al. 2010 introduced a state-of-the-art dependency tree-based classification method that uses Conditional Random Fields (CRF) with hidden variables. Dong et al. 2015 presented a statistical parsing framework for sentence-level sentiment classification. Unlike previous works that employ syntactic parsing results for sentiment analysis, they develop a statistical parser to directly analyze the sentiment structure of a sentence.

Several classifiers have been tested in our HWW2V approach, where the SVM model with RBF kernel gave the best results in terms of accuracy, i.e. 79.6%, improving further the accuracy obtained by Weighted Word2Vec (for vectors with 300 dimensions) and BoW representation. Based on our results, HWW2V methodology outperforms most of the compared approaches (Table ), and it is of low cost in terms of computational time compared to deep architectures, something of a vital interest for practical applications. In future work we plan to apply the HWW2V methodology in more datasets and explore its accuracy and efficiency to different languages, e.g. Greek, in order to further examine its value.

## Acknowledgements

This work has been supported by mSensis S.A. and the General Secretariat for Research and Technology (GSRT), Programme for the Development of Industrial Research and Technology-PAVET, *Deep Learning Methods in Sentiment Analysis* (Ref. No. 1493-BET-2013). The ownership of all possible future IPRs, related in any way with this work, belong solely to mSensis S.A. as it is stated to the respective contract between mSensis S.A. and GSRT. Authors do not claim ownership to any possible future IPRs that might be related in any way with this work.



- 
- [1] B. Pang, L. Lee, and S. Vaithyanathan, in *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*, EMNLP '02 (Association for Computational Linguistics, 2002) pp. 79–86.
- [2] B. Liu, *Synthesis Lectures on Human Language Technologies* **5**, 1 (2012).
- [3] T. Xu, Q. Peng, and Y. Cheng, *Knowledge-Based Systems* **35**, 279 (2012).
- [4] M. Hagenau, M. Liebmann, and D. Neumann, *Decision Support Systems* **55**, 685 (2013).
- [5] I. Maks and P. Vossen, *Decision Support Systems* **53**, 680 (2012).
- [6] P. D. Turney, in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02 (Association for Computational Linguistics, 2002) pp. 417–424.
- [7] V. Hatzivassiloglou and K. R. McKeown, in *Proceedings of the 35th annual meeting of the association for computational linguistics and eighth conference of the european chapter of the association for computational linguistics* (Association for Computational Linguistics, 1997) pp. 174–181.
- [8] J. Wiebe, in *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence* (AAAI Press, 2000) pp. 735–740.
- [9] M. Hu and B. Liu, in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '04 (ACM, 2004) pp. 168–177.
- [10] M. Taboada, C. Anthony, and K. Voll, in *Proceedings of the Fifth International Conference on Language Resources and Evaluation*, LREC '06 (2006) pp. 427–432.
- [11] F. Benamara, C. Cesarano, A. Picariello, D. R. Recupero, and V. S. Subrahmanian, in *Proceedings of International Conference on Weblogs and Social Media*, ICWSM '10 (2007).
- [12] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede, *Computational linguistics* **37**, 267 (2011).
- [13] R. M. Tong, in *Working Notes of the ACM SIGIR 2001 Workshop on Operational Text Classification*, Vol. 1 (2001) p. 6.
- [14] P. D. Turney and M. L. Littman, *ACM Transactions on Information Systems* **21**, 315 (2003).
- [15] C. Whitelaw, N. Garg, and S. Argamon, in *Proceedings of the 14th ACM international conference on Information and knowledge management*, CIKM '05 (ACM, 2005) pp. 625–631.
- [16] J. R. Martin and P. R. White, *The Language of Evaluation: Appraisal in English* (Palgrave Macmillan, New York, 2005).
- [17] Q. Ye, Z. Zhang, and R. Law, *Expert Systems with Applications* **36**, 6527 (2009).
- [18] R. Prabowo and M. Thelwall, *Journal of Informetrics* **3**, 143 (2009).
- [19] G. Salton and M. J. McGill, *Introduction to modern information retrieval* (McGraw-Hill, Inc., 1986).
- [20] T. Mikolov, K. Chen, G. Corrado, and J. Dean, arXiv preprint arXiv:1301.3781 (2013).
- [21] M. Wiegand, A. Balahur, B. Roth, D. Klakow, and A. Montoyo, in *Proceedings of the workshop on negation and speculation in natural language processing* (Association for Computational Linguistics, 2010) pp. 60–68.

- [22] T. Mikolov, W.-t. Yih, and G. Zweig, in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (Association for Computational Linguistics, 2013) pp. 746–751.
- [23] F. M. Zanzotto, I. Korkontzelos, F. Fallucchi, and S. Manandhar, in *Proceedings of the 23rd International Conference on Computational Linguistics, COLING '10* (Association for Computational Linguistics, 2010) pp. 1263–1271.
- [24] J. Mitchell and M. Lapata, *Cognitive science* **34**, 1388 (2010).
- [25] E. Grefenstette, G. Dinu, Y.-Z. Zhang, M. Sadrzadeh, and M. Baroni, arXiv preprint arXiv:1301.6939 (2013).
- [26] A. Yessenalina and C. Cardie, in *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '11* (Association for Computational Linguistics, Stroudsburg, PA, USA, 2011) pp. 172–182.
- [27] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, in *Proceedings of the conference on empirical methods in natural language processing, EMNLP '13*, Vol. 1631 (2013) p. 1642.
- [28] Q. V. Le and T. Mikolov, in *Proceedings of the 31st International Conference on Machine Learning, ICML '14* (2014) pp. 1188–1196.
- [29] P. J. Stone, D. C. Dunphy, and M. S. Smith, *The General Inquirer: A Computer Approach to Content Analysis* (MIT press, 1966).
- [30] A. Esuli and F. Sebastiani, in *Proceedings of the Fifth International Conference on Language Resources and Evaluation, LREC '06* (European Language Resources Association, 2006) pp. 417–422.
- [31] K. Nigam, J. Lafferty, and A. McCallum, in *IJCAI-99 workshop on machine learning for information filtering*, Vol. 1 (1999) pp. 61–67.
- [32] S. D. Pietra, V. D. Pietra, and J. Lafferty, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **19**, 380 (1997).
- [33] C. Cortes and V. Vapnik, *Machine learning* **20**, 273 (1995).
- [34] <http://www.cs.cornell.edu/people/pabo/movie-review-data/rt-polaritydata.tar.gz>.
- [35] B. Pang and L. Lee, in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, ACL '05* (Association for Computational Linguistics, 2005) pp. 115–124.
- [36] <http://scikit-learn.org>.
- [37] Y. Kim, in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (Association for Computational Linguistics, Doha, Qatar, 2014) pp. 1746–1751.
- [38] R. Socher, J. Pennington, E. H. Huang, A. Y. Ng, and C. D. Manning, in *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '11* (Association for Computational Linguistics, 2011) pp. 151–161.
- [39] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng, in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*,

- EMNLP-CoNLL '12 (Association for Computational Linguistics, 2012) pp. 1201–1211.
- [40] K. M. Hermann and P. Blunsom, in *In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics* (2013) pp. 894–904.
  - [41] S. Wang and C. D. Manning, in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ACL '12 (Association for Computational Linguistics, 2012) pp. 90–94.
  - [42] S. Wang and C. Manning, in *Proceedings of the 30th International Conference on Machine Learning*, ICML '13 (2013) pp. 118–126.
  - [43] T. Nakagawa, K. Inui, and S. Kurohashi, in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (Association for Computational Linguistics, 2010) pp. 786–794.
  - [44] L. Dong, F. Wei, S. Liu, M. Zhou, and K. Xu, *Computational Linguistics* **41**, 293 (2015).
  - [45] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, arXiv preprint arXiv:1207.0580 (2012).